

04/21/99



JC498 U.S. PTO

Please type a plus sign (+) inside this box → ☐Approved for use through 09/30/00. OMB 0651-0032
Patent and Trademark Office: U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.**UTILITY
PATENT APPLICATION
TRANSMITTAL**

(Only for new nonprovisional applications under 37 CFR 1.53(b))

| | |
|--|--|
| Attorney Docket No. | 081862.P122 |
| First Inventor or Application Identifier | Jerome A. Mouton, Jr. |
| Title | METHOD AND APPARATUS FOR UPGRADING A DATABASE IN A |
| Express Mail Label No. | EM560883474US |

APPLICATION ELEMENTS
See MPEP chapter 600 concerning utility patent application contents**ADDRESS TO:** Assistant Commissioner for Patents
Box Patent Application
Washington, DC 20231

1. ☒ Fee Transmittal Form
(Submit an original, and a duplicate for fee processing)
2. ☒ Specification *Total Pages* **27**
(preferred arrangement set forth below)
- Descriptive title of the invention
 - Cross References to Related Applications
 - Statement Regarding Fed sponsored R & D
 - Reference to Microfiche Appendix
 - Background of the Invention
 - Brief Summary of the Invention
 - Brief Description of the Drawings (if filed)
 - Detailed Description
 - Claim(s)
 - Abstract of the Disclosure

3. ☒ Drawing(s) (35 CFR 113) *Total Sheets* **8**
4. Oath or Declaration *Total Pages* ☐
- a. ☐ Newly executed (original copy)
 - b. ☐ Copy from a prior application (37 CFR 1.63(d))
(for continuation/divisional with Box 16 completed)
[Note Box 5 below]
 - i. ☐ **DELETION OF INVENTOR(S)**
Signed statement attached deleting
inventor(s) named in the prior application,
see 37 CFR 1.63(d)(2) and 1.33(b).

NOTE FOR ITEMS 1 & 13: IN ORDER TO BE ENTITLED TO PAY SMALL ENTITY FEES, A SMALL ENTITY STATEMENT IS REQUIRED (37 C.F.R. § 1.27), EXCEPT IF ONE FILED IN A PRIOR APPLICATION IS RELIED UPON (37 C.F.R. § 1.28).

5. ☐ Microfiche Computer Program (Appendix)
6. Nucleotide and/or Amino Acid Sequence Submission
(if applicable, all necessary)
- a. ☐ Computer Readable Copy
 - b. ☐ Paper Copy (identical to computer copy)
 - c. ☐ Statement verifying identity of above copies

ACCOMPANYING APPLICATION PARTS

7. ☐ Assignment Papers (cover sheet & document(s))
8. ☐ 37 CFR 3.73(b) Statement ☐ Power of Attorney
(when there is an assignee)
9. ☐ English Translation Document (if applicable)
10. ☐ Information Disclosure Statement (IDS)/PTO - 1449 ☐ Copies of IDS Citations
11. ☐ Preliminary Amendment
12. ☒ Return Receipt Postcard (MPEP 503)
(Should be specifically itemized)
13. ☐ *Small Entity ☐ Statement filed in prior application,
Statement(s) ☐ Status still proper and desired
14. ☐ Certified Copy of Priority Document(s)
(if foreign priority is claimed)
15. ☐ Other:

16. If a CONTINUING APPLICATION, check appropriate box, and supply the requisite information below and in a preliminary amendment:☐ Continuation ☐ Divisional ☐ Continuation-in-part (CIP) of prior application No. _____ / _____

Prior application Information: Examiner _____ Group/Art Unit: _____

For CONTINUATION or DIVISIONAL APPS only: The entire disclosure of the prior application, from which an oath or declaration is supplied under Box 4b, is considered a part of the disclosure of the accompanying continuation or divisional application and is hereby incorporated by reference. The incorporation can only be relied upon when a portion has been inadvertently omitted from the submitted application parts.

17. CORRESPONDENCE ADDRESS☐ Customer Number of Bar Code Label

(Insert Customer No. or Attach bare code label here)

or ☒ Correspondence address below

| | | | | | |
|---------|---|-----------|----------------|----------|----------------|
| Name | BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN LLP | | | | |
| Address | 12400 Wilshire Boulevard, Seventh Floor | | | | |
| City | Los Angeles | State | California | Zip Code | 90025 |
| Country | U.S.A. | Telephone | (310) 207-3800 | Fax | (310) 820-5988 |

Name (Print/Type) George G. C. Tseng; Reg. No. 41,355

Signature

Date

04/21/99

Burden Hour Statement: This form is estimated to take 0.2 hours to complete. Time will vary depending upon the needs of the individual case. Any comments on the amount of time you are required to complete this form should be sent to the Chief Information Officer, Patent and Trademark Office, Washington, DC 20231. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND TO: Assistant Commissioner for Patents, Box Patent Application, Washington, DC 20231.

Our File No: 081862.P122
Express Mail No: EM560883474US

UNITED STATES LETTERS PATENT APPLICATION

FOR

**METHOD AND APPARATUS FOR UPGRADING A DATABASE
IN A REDUNDANT ENVIRONMENT BY RELEASE CHAINING**

Inventors:

Jerome A. Mouton, Jr.
Alex V. Truong
William P. Buckley

Prepared by:

BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN LLP
12400 Wilshire Boulevard, Seventh Floor
Los Angeles, California 90025-1026
(310) 207-3800

**METHOD AND APPARATUS FOR UPGRADING A DATABASE
IN A REDUNDANT ENVIRONMENT BY RELEASE CHAINING**

Field of the Invention

5 This invention relates to database data and structure upgrades. Specifically, this invention is directed towards the upgrading of a database in a redundancy environment by release chaining.

Background

10 A database as described herein consists of one or more related tables of compound data structures. The database's schema is defined as the organization of its tables and the relationships among them. A version of such a database is a specific schema and the specific data in the structures. As they evolve over time,
15 databases are embodied in a series of versions, each with a changed schema and new data elements. A new version of the database is generated from an old one by upgrading its schema and mapping its data to the new schema. Database software will gener-
ally support upgrading from any of several previous versions.

20 Certain database systems provide fault tolerance by main- taining redundant images of a database, and using the mirrored images in case of failure of the primary database. A messaging protocol is used to keep mirror images identical to the primary database. The basic protocol is the transmission of a message
25 containing one or more database records, in a specific format, to the respective mirror databases, and commitment of the data upon receiving an acknowledgement, or backout in case of failure..

In a redundancy environment, upgrading is sometimes performed by upgrading a mirror image database to the new version and then at the appropriate time switching to use the mirror image as the primary database. In this process, upgrading is performed by receiving database update messages from a previous version and mapping them into the schema of the new version. An empty database structure conforming to the schema of the new version is created to accept these mappings.

A new release of a database software might support the upgrading of databases from up to three previous generations. In existing systems, each new software release provides separate modules for each prior release from which an upgrade is possible. Upgrade code must be developed to map each of these three old releases into the current release, with all of the upgrade code for each new version written anew in each release (using older release code as a model). Thus, if the latest database software is at release 4, and assuming that releases have been numbered only at full numeral versions (e.g., release 1, release 2 and release 3), then the upgrade code for release 4 must contain code to convert the databases to release 4 directly from release 1, release 2, or release 3.

The current upgrade system requires the duplication of code for mapping and writing databases for each prior supported version. This also increases complexity in code as analysis must be done between any prior release and the newest release to understand how different versions must be changed to arrive at the latest release. For example, developing the release 2 to release

SUMMARY

It is therefore an object of the present invention to reduce the amount of code development necessary in developing a new release of a database system and operating code.

5 It is a further object of the present invention to reduce the amount of code redundancy from one version of the operating code as compared to an earlier version.

These and other objects of the invention are provided by a system that provides for version upgrades of the database to be processed in a chained manner. Update protocol messages from a previous version are mapped into protocol messages of the next most recent version. These are mapped, in turn, to a more recent version until the message of the current version has been derived. Then, the current version update message is processed as in a normal database update to write the current version of the database.

Other objects, features, and advantages of the present invention will be apparent from the accompanying drawings and from the detailed description which follows below.

BRIEF DESCRIPTION OF THE DRAWINGS

The system is illustrated by way of example and not limitation in the figures of the accompanying drawings in which like references indicate similar elements and in which:

5 **Figure 1** is a block diagram of a switch containing a set memory systems configured in accordance to one embodiment of the present invention.

Figure 2a-2d is a set of block diagrams illustrating changes in the memory systems of **Figure 1** during an upgrade performed in accordance with one embodiment of the present invention.

Figure 3 is a flow diagram illustrating a process for upgrading the memory systems in **Figures 2a-2d** in accordance with one embodiment of the present invention.

Figure 4 is a diagram illustrating the flow of update messages through a set of update and upgrade functions provided by one embodiment of the present invention to upgrade the memory systems of **Figure 2a-2d**.

Figures 5a-5e is a set of block diagrams illustrating changes in a memory system of **Figure 1** during an upgrade performed in accordance with one embodiment of the present invention.

Figure 6 is a flow diagram illustrating a process for upgrading the memory system in **Figures 5a-5e** in accordance with one embodiment of the present invention.

25 **Figure 7** is a diagram illustrating the flow of update messages through a set of update and upgrade functions provided by

one embodiment of the present invention to upgrade the memory systems of **Figure 5a-5e**.

SECRET

DETAILED DESCRIPTION

In order to reduce code redundancy and possibilities for error in the generation of new code, the system provides for version upgrades of the database to be processed in a chained manner. The databases used in the present invention may be any databases for which a message-based upgrade protocol is used. Update protocol messages from a previous version are mapped into protocol messages of the next most recent version. These are mapped, in turn, to a more recent version until the message of the current version has been derived. Then, the current version update message is processed as in a normal database update to write the current version of the database.

For example, in a release 4 system, upgrading a database from release 2 involves the following sequence:

1. Receiving the release 2 update message;
2. Mapping the release 2 update message into a release 3 update message;
3. Mapping the release 3 update message into a release 4 update message; and,
4. Updating the release 4 database using the normal update code.

Benefits of release chaining include:

Reduced Code Development: Instead of developing essentially new code to map each of several old version messages into the current version, it is only necessary to develop code to map the most recent version into the current version. Older version are

processed by chaining to this new code.

Improved Quality: While upgrade code is usually performed once per customer, database update code is used every day in every system. So, much of the code for performing upgrades is much more thoroughly tested than in the old method. In addition, the "links" in the chain for older versions are identical code to that used for upgrades in earlier versions, so the upgrades also benefit from being exercised during all the upgrades performed previously. In the prior art, all of the upgrade code for each new version was basically written new in each release of software (using older release software code as a model).

Consistency: Since the update code for the current version database is used to actually write the upgraded databases, it is always written in exactly the same manner as for other operations.

Reduced Code Redundancy: Instead of duplicate code for each supported release of software for mapping and writing databases, all mapping of one version's format and data occurs in one module rather than in several places. For example, in the old method the mapping contained in release 3 software to release 4 software is also implicitly included in the mapping from release 2 software to release 4 software. With chaining, update messages conforming to release 2 software is mapped to messages conforming to release 3 software, these messages are mapped to conform to messages generated by release 4 software, then the database is updated using messages generated by release 4 update code.

Reduced Code Complexity: Developing the release 2 to release 4 upgrade code using the old method involves understanding changes

from different versions of the database supported by the different releases of software, and combining them into a single module. In addition, the complexity of mapping an update message into the database itself must also be included in each set of upgrade code.

5 In the chained method, code to map release 2 to release 3 is written once--in release 3--and only maps release 2 update messages to release 3 messages. Writing an update message to the database is always done with the update code of the current release.

6 The databases in one embodiment of the present invention are contained in a real-time network switch containing redundant processors. The databases consist of data structures needed to perform the various functions of the network switch, including the control of the switching and the maintenance of historical data for the network. The databases are stored in memory devices such as random access memory or battery-backed random access memory on the network switch. Although the invention is described in context of use in a network switching device, the invention applies to any database using a message-based update protocol.

20 **Figure 1** is a block diagram of a switch 50 containing a control card 75 with a memory system 100 configured in accordance with one embodiment of the present invention. Memory system 100 contains a random access memory (RAM) 102, a read only memory (ROM) 104, and a battery backed random access memory (BRAM) 106.

25 RAM 102 may be implemented using dynamic RAM (DRAM) or synchronous DRAM (SDRAM). ROM 104 may be implemented using flash memory or such non-volatile memory devices as magnetic and/or optical

drives. BRAM 106 is implemented as RAM that has back-up power supplied by batteries. Similar to ROM 104, BRAM 106 may also be implemented using other types of non-volatile storage medium. Depending on the performance requirements of the specific
5 implementation, RAM 102, ROM 104, and BRAM 106 may be implemented using other types of storage devices. In a preferred embodiment, RAM 102 and BRAM 106 must be implemented with storage devices that are writable.

Memory system 100 contains the program code and data necessary in the operation of switch 50. Switch 50 uses RAM 102 to store the running code and the active configuration databases. The databases include routing tables for the network to which switch 50 is connected, status tables to hold the status of the cards in switch 50, and the other tables used in the operation of switch 50.

If switch 50 fails (e.g., switch 50 suffers a power loss or is reset), RAM 102 is cleared as it becomes un-powered. ROM 104 is then used to recover the program code in RAM 102 once switch 50 is operational again as ROM 104 is a non-volatile memory. BRAM
20 106 contains a back-up of the set databases. This back-up set of databases is continuously maintained and updated by the executing code in RAM 102. Even if switch 50 fails, RAM 102 may be refreshed by the data retrieved from BRAM 106. RAM 102 is used in the execution of code and storage of the active configuration
25 databases as RAM 102 is generally faster in operation than ROM 104 and BRAM 106.

Coupled to memory system 100 is a processor 108, which is a

processor that executes the code in memory system 100 during the operation of switch 50. Processor 108 is also coupled to a networking switching device 110. Network switching device 110 controls the switching of data for a set of communication interface cards 112. Communication interface cards 112 send and receive data from a network (not shown). Memory system 100, processor 108, and network switching device 110 are contained on controller card 75.

Switch 50 also contains a second controller card 575 that acts as a back-up for controller card 75. Thus, in case controller card 75 suffers an error in software or fails in hardware, second controller card 575 can take over and operation in switch 50 can continue with the minimal of interruption. Controller card 575 contains a memory system 500 that acts in the exact manner as memory system 100 does for controller card 75. Memory system 500 contains a RAM 502 (for storing executing operating code and databases), a ROM 504 (for storing an image of the operating code), and a BRAM 506 (for storing a back-up of the databases). Memory system 500 is coupled to a processor 508 and a network switching device 510. The description to memory system 100, processor 108 and network switching device 110 are also applicable to memory system 500, processor 508 and network switching device 510 in terms of operation and function. As the standby card, second controller card 575 maintains a backup of the databases and operating code in memory system 500 that is contained in memory system 100. The databases in memory system 500 of controller card 575 are updated by using update messages

generated from the databases in memory system 100 of controller card 75.

Figures 2a-2d is a series of diagrams illustrating the changes in the software contained in memory system 100 and memory system 500 during an upgrade. **Figures 2a-2d** is described with reference to **Figure 3**, which is a flow diagram illustrating the sequence of events in the upgrade process. Generally, the process involves loading an image of the new release of operating code into ROM 104 and RAM 502. The process continues with creating new database structures in RAM 502, which conform to the specifications of the latest version, and updating the structures with update messages generated from the databases in RAM 102. The databases in BRAM 506, which still conform to the schema of the old version, are also updated as a safeguard such that controller card 575 can still be used as an active card.

The latest mapping functions are hard-coded into the software image of the new release. All the upgrade processing is performed through execution of the new release code. The "old versions" of the mappers are compiled and used in the new release software module, but the object code of these mappers is not the same as when they were compiled for the old release because of different addresses and offsets in the new release. So, the source of those mappers is the same in the new release, but the executable code is quite different.

Initially, the update messages that are generated from the databases in RAM 102 conform to the older version of the database and cannot be directly used to update the databases in RAM 502 as

the databases in RAM 502 are of a newer, different version. Thus, the format and content of the update messages are first upgraded to the latest version (e.g., the version of the databases in RAM 502), using one or more intermediary mappers before the update message can be used to update the databases in RAM 502. The update messages for BRAM 506 still require the old version of the update messages as the version of the databases in BRAM 506 are the same as the version of the databases in RAM 102. After the databases in RAM 502 have been completely populated through the use of upgraded update messages and control has been switched over to controller card 575 from controller card 75, the databases in RAM 502 are copied to BRAM 506, replacing the prior version databases in BRAM 506.

Figure 2a shows the release numbers of the software in RAM 102, ROM 104, and BRAM 106 of memory system 100 and in RAM 502, ROM 504, and BRAM 506 of memory system 500 before the upgrade. RAM 102 and RAM 502 contain the operating code and the active databases, both of which are at version 8.5.00. ROM 104 and ROM 504 contain a non-volatile copy of the code, which is also at version 8.5.00. BRAM 106 and BRAM 506 contain the battery backed copies of the databases, which are at version 8.5.00. For purposes of explanation, it is assumed that controller card 75 is the active controller card and controller card 575 is the standby controller card.

Referring to **Figure 2b** and block 600 of **Figure 3**, the active ROM and the standby RAM (e.g., ROM 104 and RAM 502, respectively), are loaded with an image of the updated release of

the operating code, version 9.2.00, using a protocol such as the trivial file transfer protocol (TFTP). The loaded image contains all code needed for the operation of switch 50, which includes code needed to upgrade the databases contained in switch 50. The configuration databases in RAM 502 are cleared by the loading of the new software image. The old version of the databases are flushed from RAM 502 because these databases are not fully compatible with the new software. As described above, incompatibilities between the old versions of the databases and the new operating code are due to changes in the schema of the new databases and changes in the new operating code to operate with the new formats.

After ROM 104 and RAM 502 are loaded with the new release of the code, any reset or re-initialization of switch 50 results in the standby controller card, controller card 575, becoming the active controller card and loading the old release operating code image from ROM 504. Controller card 575 then rebuilds the databases in RAM 502 using the old version of the databases contained in BRAM 506. Controller card 575 becomes the active controller card as the ROM in controller card 75, which contains the operating code for the new version of the database has not been specified by the user as the primary version.

As shown in **Figure 2c** and in block 602 of **Figure 3**, the new version of the databases in RAM 502 is updated with update messages from RAM 102. As described below, the update is performed by chaining mappings from one version to the next. After block 602, the upgraded databases in RAM 502 have been

updated with the most current information from RAM 102. As the
databases in RAM 502 are being updated using chain mapped updates,
the databases in BRAM 506 is also constantly being updated using
the old version of the update messages. This is because the
5 databases in RAM 102 and BRAM 506 are the same version.

In block 604, the user issues a command to switch 50 to run
the new revision. This causes control of switch 50 to pass from
the active controller card, controller card 75, to the new version
upgraded controller card, controller card 575. When controller
10 card 575 receives control and thus becomes the now active
controller card, the software on controller card 575 rechecks
local configuration and completes integration of the new
databases. This occurs with little or no impact to data
transmission through the switch. In the preferred embodiment, the
15 switchover to controller card 575 occurs at user control through a
command issued by the user and does not take place automatically.

In block 606, as the databases in BRAM 506 can no longer be
used as the databases in BRAM 506 is an older version, the system
copies the databases in RAM 502 to BRAM 506, replacing the
20 databases in BRAM 506 with the latest version. This is shown in
Figure 2d. The new version databases are not rewritten to BRAM
until after the switchover has been requested and completed in a
satisfactory manner to allow the controller card 575 to still act
as an active controller card in case there is a problem with the
25 upgrade. After the databases have been copied to BRAM 506 from
RAM 502, ROM 504 is also upgraded with the new release of the
operating code image and operation continues as normal with the

upgraded software.

Figure 4 illustrates the function call and associated data flow for the chained upgrade process used in block 602. The upgrade, which is initiated by a user, begins when the system calls a db_update function 702. Db_update function 702 is directed to update the databases in RAM 502 by using a release 8.5 update message (Rel 8.5 Update) 714.

The first mapping in the upgrade sequence uses a release 8.5 update to RAM mapper (Rel 8.5 u2r mapper) 704 to build a release 9.1 update message (Rel 9.1 Update) 716 from Rel 8.5 Update 714. Rel 8.5 u2r mapper 704 calls a release 8.5 to release 9.1 update to update mapper (Rel 8.5->9.1 u2u mapper) 710 to generate Rel 9.1 Update 716.

After Rel 9.1 Update 716 has been created, a release 9.1 update to RAM mapper (Rel 9.1 u2r mapper) 706 is called to create a release 9.2 update message (Rel 9.2 Update) 718. A release 9.1 to 9.2 update to update message mapper (Rel 9.1->9.2 u2u mapper) 712, which is a function similar to Rel 9.1->9.2 u2u mapper 712, is called by Rel 9.1 u2r mapper 706 to create Rel 9.2 Update 718.

Assuming that there are no further mappings necessary in the upgrade process after Rel 9.1->9.2 u2u mapper 412 is executed, the last part of the mapping sequence is to call the current update to RAM mapper (Current u2r mapper) 708 to write the Rel 9.2 Update 718 to the databases in RAM 502.

Whether there is one or several update messages generated for each database to be updated is a design decision by the implementers of each database. In some cases one update message

is used for the whole database (e.g., the update message contains all the database records for a database), in others the database is sent over in several update messages (e.g., the update message contains a portion of the database records for a database). The decision usually depends on the size and structure of the database, with consideration for the need to have as little loss of data as possible. In addition, whether the schema of a database in the set of databases changes between different releases of software is also an implementation decision. Thus, a new release of the software image does not necessarily mean that there is a new version of a database schema for any of the databases. In the explanation given herein, however, it is assumed that the databases schemas are changed with each release. Specifically, a different release of the software image has a different version of the database schema for at least one database.

Figures 5a-5e is a series of diagrams illustrating the changes in the software contained in memory system 100 during an upgrade where switch 50 only contains a single controller card, controller card 75. **Figures 5a-5e** is described with reference to **Figure 6**, which is a flow diagram illustrating the sequence of events in the upgrade process. Generally, the process involves loading an image of the new release of operating code into ROM 104 and RAM 102. The process continues with creating new database structures in RAM 102, which conform to the specifications of the latest version, and updating the structures using update messages generated from the databases in BRAM 106. Initially, the update

messages that are generated from the databases in BRAM 106 conform to the older version of the database and cannot be directly used to update the databases in RAM 102 as the databases in RAM 102 are of a newer, different version. Thus, the format and content of the update messages are first upgraded to the latest version (i.e., the version of the databases in RAM 102) using one or more intermediary mappers before the update message can be used to update the databases in RAM 102. After the databases in RAM 102 have been completely populated through the use of upgraded update messages, the databases in RAM 102 are copied to BRAM 106, replacing the prior version databases in BRAM 106.

Figure 5a shows the release numbers of the software in RAM 102, ROM 104, and BRAM 106 of the memory system 100 before the upgrade. As described above, RAM 102 contains the operating code and the active databases, both of which are at version 9.1.05. ROM 104 contains a non-volatile copy of the code, which is also at release 9.1.05. BRAM 106 contains the battery backed copies of the configuration databases, which are at release 9.1.05.

Referring to **Figure 5b** and block 300 of **Figure 6**, ROM 104 is loaded with an image of the updated release of the operating code, version 9.2.00, using a protocol such as the trivial file transfer protocol (TFTP). After ROM 104 is loaded with the new release of the code, any reset or re-initialization of switch 50 will cause RAM 102 to be cleared and the code in ROM 104 to be loaded into RAM 102. As described below, the code in ROM 104 is loaded into RAM 102 in block 302, when the user issues an upgrade command.

In block 302 of **Figure 6**, the new release of the code, version 9.2.00, in ROM 104 is loaded into RAM 102. At this point, the configuration databases in RAM 102 are cleared by the loading of the new software image. The old version of the databases are
5 flushed from RAM 102 because these databases are not fully compatible with the new software. Incompatibilities between the old versions of the databases and the new operating code are due to changes in the formats of the new databases and changes in the new operating code to operate with the new formats. During the
10 upgrade of switch 50, significant service outage in switch 50 occurs and may last from minutes to hours, depending on the amount of provisioning. Outage occurs due to the lack of a redundant controller card in this embodiment of switch 50 to continue operation when all code and databases in RAM 102 are cleared after
15 the uploading of the image of the new software. The state of memory system 100 after block 302 has completed is shown in **Figure 5c**. Nothing of the old release of the software image is preserved in switch 50 past the loading of the new image into RAM 102-- except for the configuration databases in BRAM 106.

20 As shown in **Figure 5d** and in block 304 of **Figure 6**, the new version of the databases in RAM 102 is updated with update messages from BRAM 106. As described below, the update is performed by chaining mappings from one version to the next. After block 304, the upgraded databases in RAM 102 have been
25 updated with the most current information from BRAM 106. Thus, in block 306, as the databases in BRAM 106 can no longer be used as the databases in BRAM 106 is an older version, the system copies

the databases in RAM 102 to BRAM 106, replacing the databases in
BRAM 106 with the latest version. This is shown in **Figure 5e**.
After the databases have been copied to BRAM 106 from RAM 102,
operation of control card 75 continues as normal with the upgraded
5 software.

Figure 7 illustrates the function call and associated data
flow for the chained upgrade process used in block 304. The
upgrade, which is initiated by a user, begins when the system
calls a rebuild_n_recover function 402. Rebuild_n_recover
10 function 402 recreates the databases in RAM 102 by using update
messages.

The first mapping in the upgrade sequence uses a release 9.1
BRAM to RAM mapper (Rel 9.1 b2r mapper) 404 to build a release 9.1
update message (Rel 9.1 Update) 414 from the databases in BRAM
106. Rel 9.1 b2r mapper 404 performs the entire function of
15 moving the contents of the databases in BRAM 106 to RAM 102 by
invoking a release 9.1 BRAM to Update mapper (Rel 9.1 b2u mapper)
410 to generate Rel 9.1 Update 414. In contrast to the mappings
needed for updates in a scenario where switch 50 contains two
20 controller cards, Rel 9.1 b2u mapper 410 is the additional mapping
to be performed for an upgrade where switch 50 contains only a
single controller card.

After Rel 9.1 Update 414 has been created, a release 9.1
update to RAM mapper (Rel 9.1 u2r mapper) 406 is called to create
25 a release 9.2 update message (Rel 9.2 Update) 416. A release 9.1
to 9.2 update to update message mapper (Rel 9.1->9.2 u2u mapper)
412, which is a function similar to Rel 9.1 b2u mapper 410, is

called by Rel 9.1 u2r mapper 406 to create Rel 9.2 Update 416.

Additional mappers may be used in cases where there are more conversions necessary to convert the update message to the latest revision. For example, if the version to be upgraded to were actually release 9.3 instead of release 9.2, then an additional mapping stage would be required. This mapping may be performed by a function such as a release 9.2->9.3 u2u mapper. Assuming that there are no further mappings necessary in the upgrade process after Rel 9.1->9.2 u2u mapper 412 is executed, the last part of the mapping sequence is to call the current update to ram mapper (Current u2r mapper) 408 to write Rel 9.2 Update 416 to the databases in RAM 102.

Figures 5-7 describe the situation where controller card 75 is the only controller card in switch 50. In many mission critical applications, however, a desired configuration is to have a second controller card (e.g., controller card 575) in switch 50 to act as a standby controller card. If there is a failure in controller card 75, switch 50 can change over to the second controller card with very little loss of data. To ensure the least amount of latency for transfer, the second, standby, controller card is continuously updated with information from memory system 100 to closely replicate the contents of memory system 100. Unlike the upgrade of a switch containing only one controller card (where significant service outage in the switch occurs and may last from minutes to hour), the upgrade of switch 50 containing a second controller card does not cause significant service outage. In many cases, users might not even be aware when

an upgrade has been performed. The minimization of the outage is achieved by the redundant controller card in this embodiment of switch 50 to continue operation while all code and databases in RAM 502 are upgraded and updated.

5 In the foregoing specification, the invention has been described with reference to specific exemplary embodiments thereof. It will, however, be evident that various modifications and changes may be made thereto without departing from the broader spirit and scope of the invention as set forth in the appended claims. The specification and drawings are, accordingly, to be regarded in an illustrative rather than a restrictive sense.

10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218

CLAIMS:

What is claimed is:

- 1 1. A method of updating a message from a first version to an
2 upgraded version by chaining through intermediate versions
3 comprising:
4 receiving an update message having a first version format;
5 and,
6 repeatedly generating a revised update message having a next
7 most recent version format based on the update message until a
8 final update message having an upgraded version format is
9 generated.
- 1 2. The method of claim 1, wherein generating a revised update
2 message having a next most recent version format includes:
3 receiving a first update message; and,
4 calling a next most recent version mapping function to map
5 contents of the first update message to generate a second update
6 message.
- 1 3. The method of claim 1, wherein the update message includes a
2 set of records for a database in the first version.
- 1 4. The method of claim 3, wherein the set of records for the
2 database in the first version is a complete set of records for the
3 database.
- 1 5. An article comprising a computer readable medium having
2 instructions stored thereon, which when executed, causes:

3 receipt of an update message having a first version format;
4 and,
5 repeated generation of a revised update message having a next
6 most recent version format based on the update message until a
7 final update message having an upgraded version format is
8 generated.

1 6. The article of claim 5, wherein the computer readable medium
2 further having instructions stored thereon, which when executed,
3 causes:

4 receipt of a first update message; and,
5 calling of a next most recent version mapping function to map
6 contents of the first update message to generate a second update
7 message.

1 7. The article of claim 5, wherein the update message includes a
2 set of records for a database in the first version.

1 8. The article of claim 7, wherein the set of records for the
2 database in the first version is a complete set of records for the
3 database.

1 9. An apparatus for updating a message from a first version to
2 an upgraded version by chaining through intermediate versions
3 comprising:

4 means for receiving an update message having a first version
5 format; and,

6 means for repeatedly generating a revised update message
7 having a next most recent version format based on the update

8 message until a final update message having an upgraded version
9 format is generated.

1 10. The apparatus of claim 9, further comprising:
2 means for receiving a first update message; and,
3 means for calling a next most recent version mapping function
4 to map contents of the first update message to generate a second
5 update message.

1 11. The apparatus of claim 9, wherein the update message includes
2 a set of records for a database in the first version.

3 12. The apparatus of claim 11, wherein the set of records for the
4 database in the first version is a complete set of records for the
5 database.

ABSTRACT

What is disclosed is a method of updating a message from a first version to an upgraded version by chaining through intermediate versions, including the steps of receiving an update message having a first version format, and repeatedly generating a revised update message having a next most recent version format based on the update message until a final update message having an upgraded version format is generated. An apparatus for performing the method is also disclosed.

RECEIVED

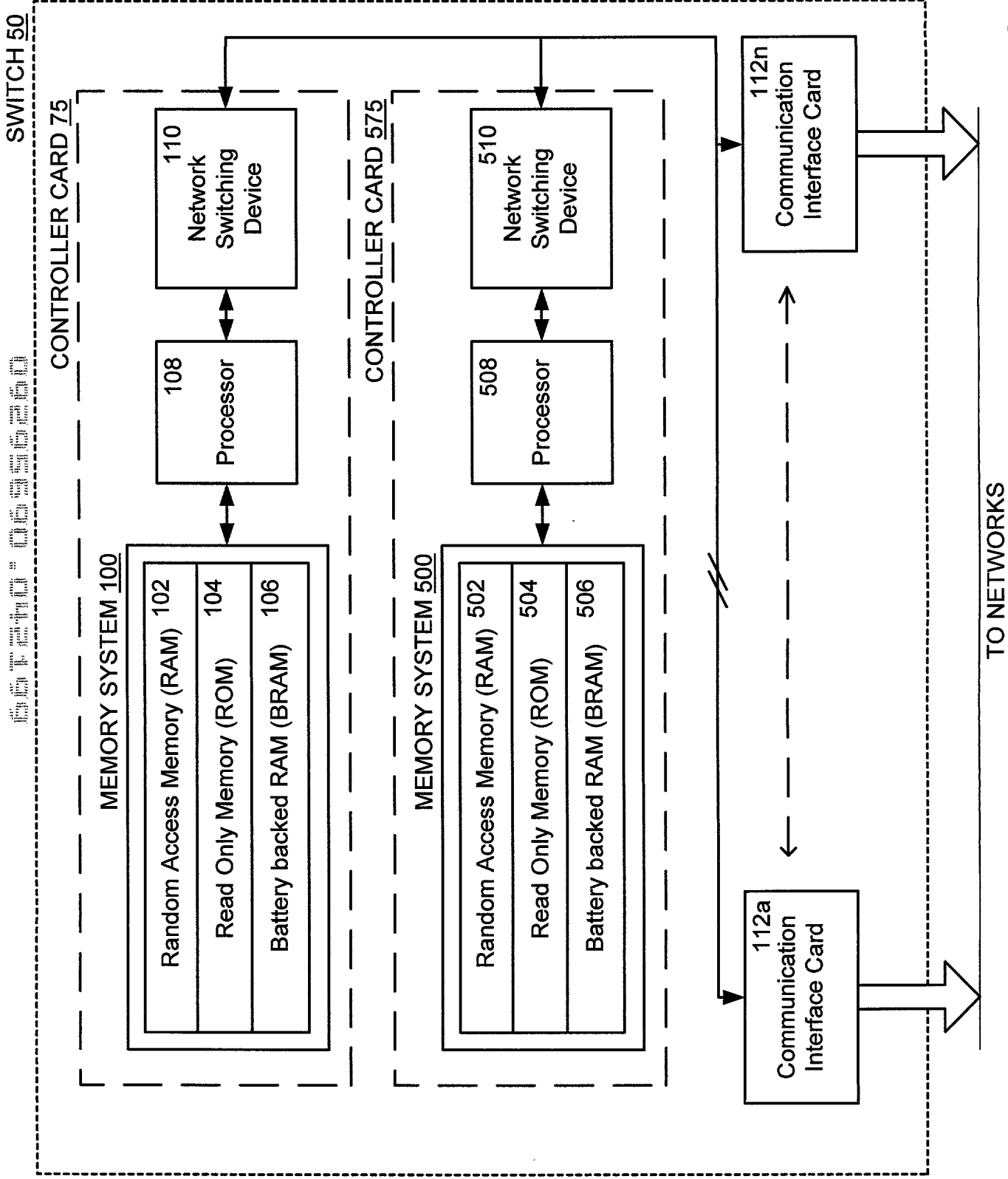


Fig. 1

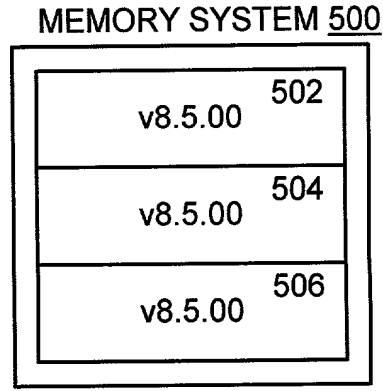
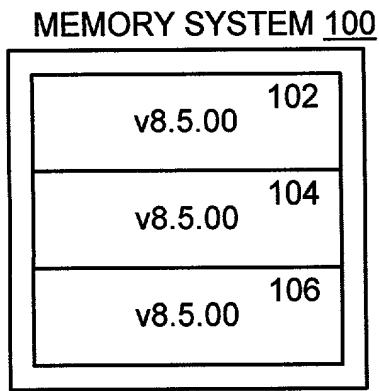


Fig. 2a

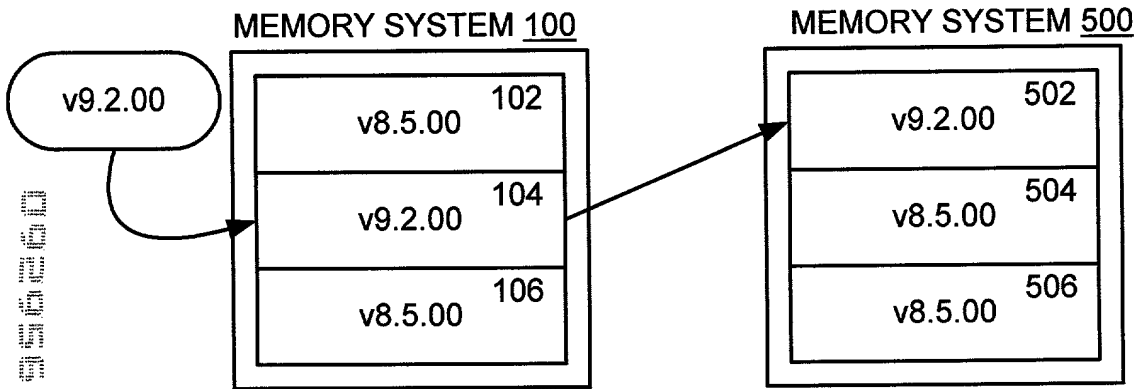


Fig. 2b

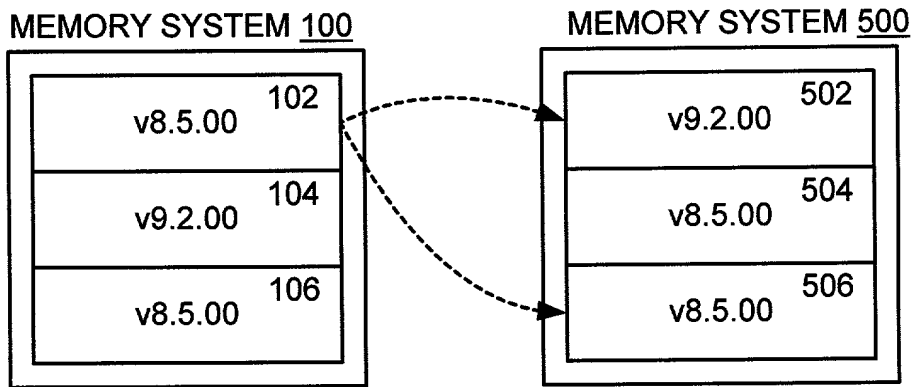


Fig. 2c

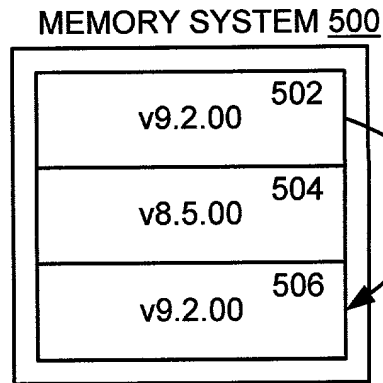
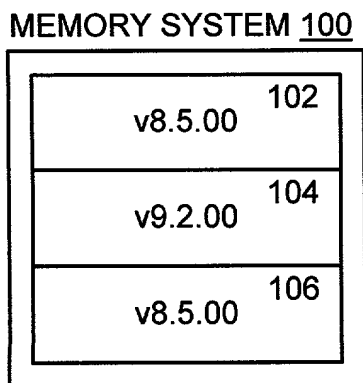


Fig. 2d

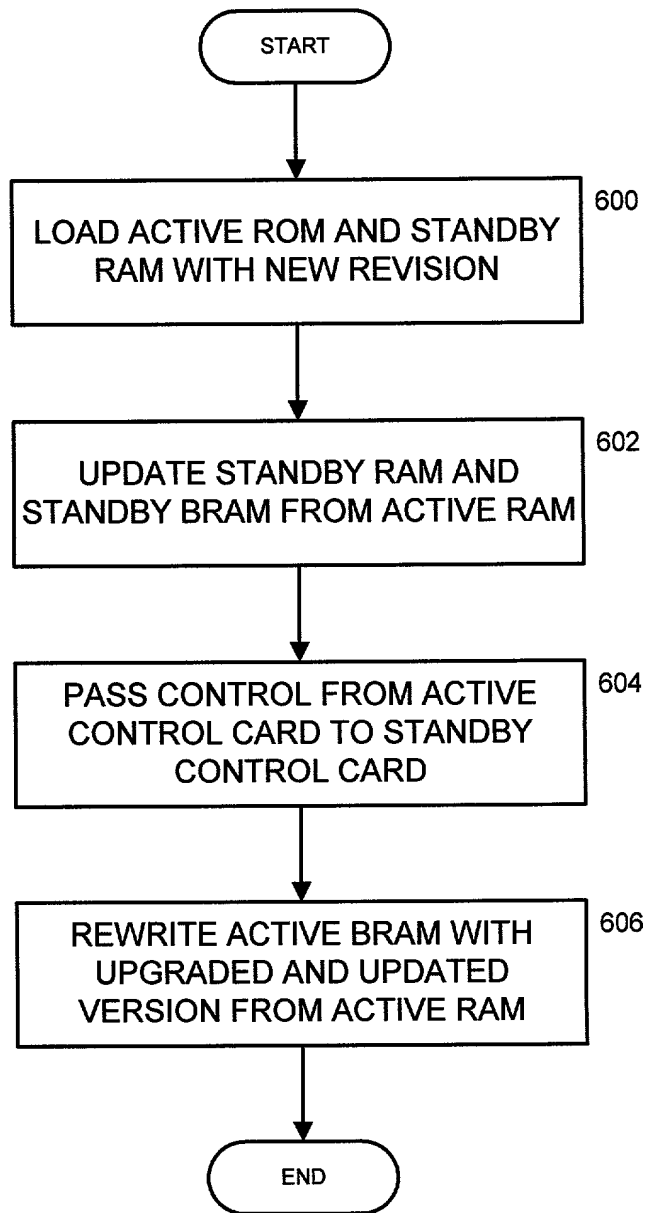


Fig. 3

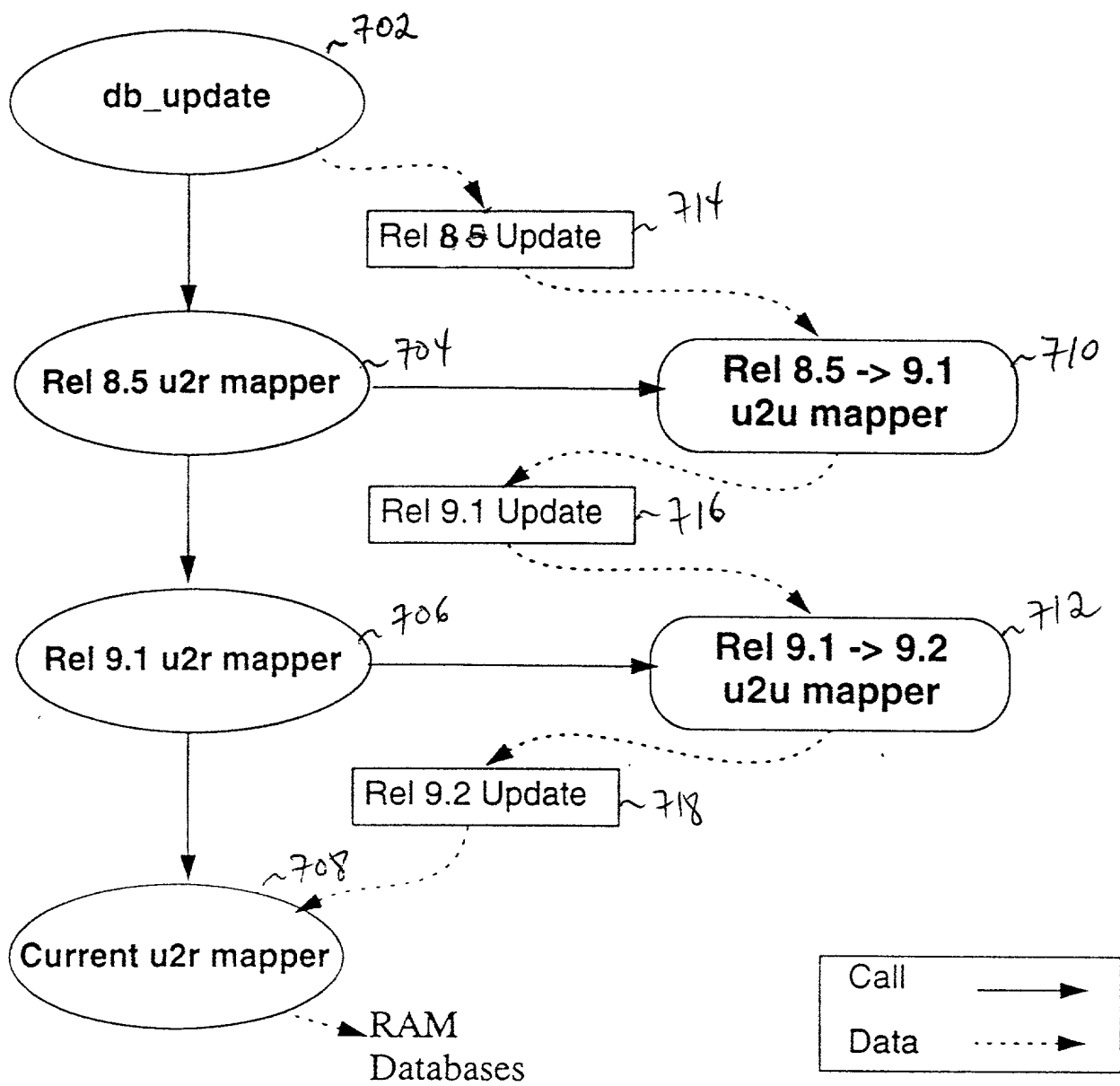


FIG. 4

MEMORY SYSTEM 100

| | |
|---------|-----|
| v9.1.05 | 102 |
| v9.1.05 | 104 |
| v9.1.05 | 106 |

Fig. 5a

MEMORY SYSTEM 100

| | | |
|---------|---------|-----|
| v9.2.00 | v9.1.05 | 102 |
| | v9.2.00 | 104 |
| | v9.1.05 | 106 |

Fig. 5b

MEMORY SYSTEM 100

| | |
|---------|-----|
| v9.2.00 | 102 |
| v9.2.00 | 104 |
| v9.1.05 | 106 |

Fig. 5c

MEMORY SYSTEM 100

| | |
|---------|-----|
| v9.2.00 | 102 |
| v9.2.00 | 104 |
| v9.1.05 | 106 |

Fig. 5d

MEMORY SYSTEM 100

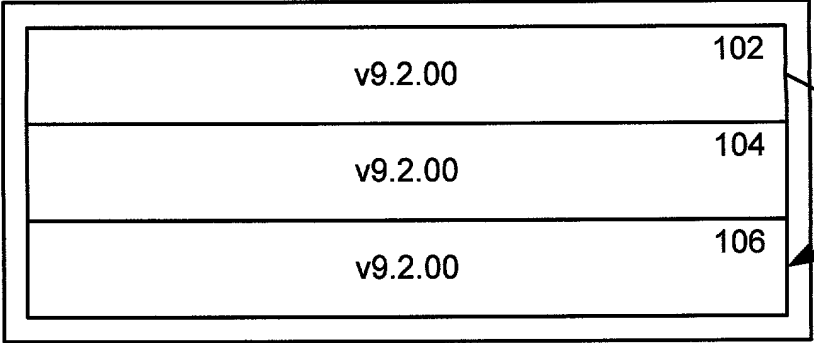


Fig. 5e

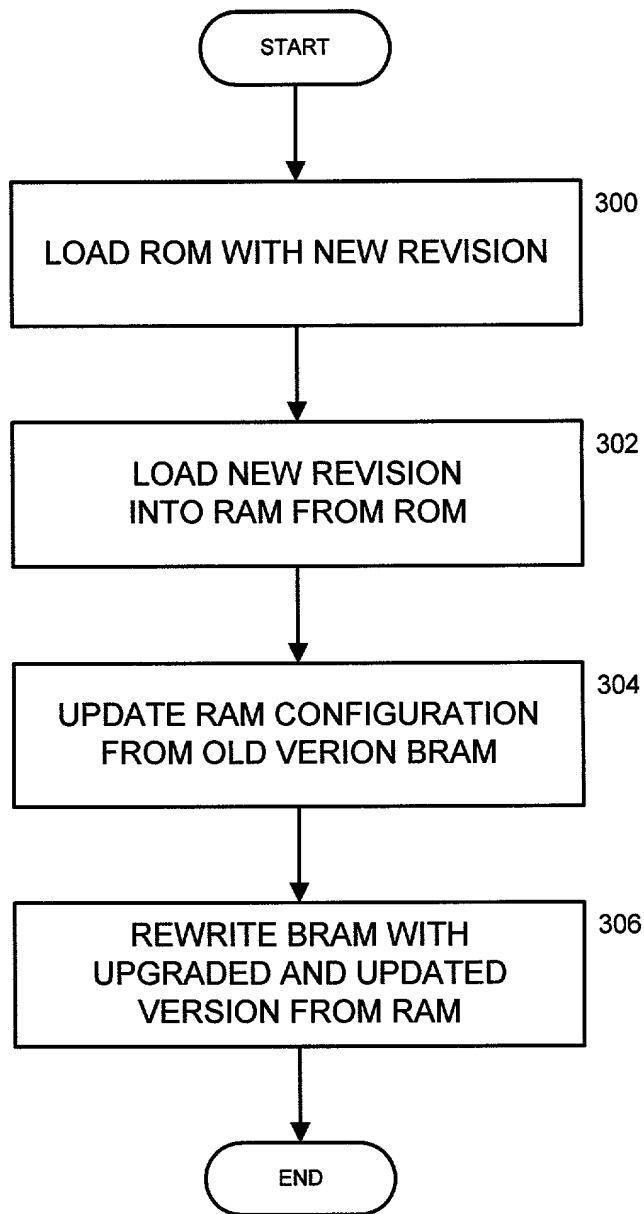


Fig. 6

